

# Ice Sheet System Model

## Uncertainty quantification of PIG

Nicole SCHLEGEL<sup>1</sup>,

<sup>1</sup>California Institute of Technology - JPL, Pasadena, CA



# Overview

① Introduction

② Sampling Analysis

③ Sensitivity Analysis

④ Plot results

⑤ Conclusion

⑥ Additional Exercises

## Introduction

Follow on study on Pine Island Glacier, to assess how errors in model inputs propagate through a 2D SSA steady-state ice flow model.

**Our model inputs:** ice thickness, ice rigidity and basal friction.

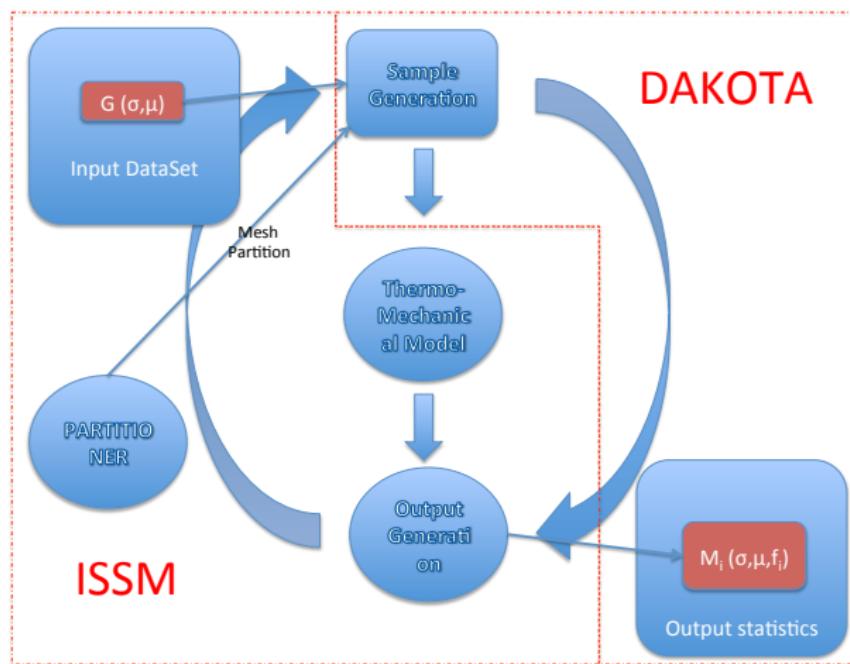
**Our model outputs:** mass flux at 13 flux gates across PIG.

Steps:

- Load 09 PIG Application results (starting from the end of the basal friction inversion)
- Load ice thickness cross-over errors from IceBridge 2009 WAIS campaign
- Run sampling analysis using ice thickness cross-over and mass flux diagnostics.
- Run sensitivity analysis using ice thickness, ice rigidity and basal friction as inputs and mass flux diagnostics
- Plot results: partition
- Plot results: sampling
- Plot results: sensitivities

*See [Larour et al, 2012 b and c] for more details on ISSM QMU capabilities*

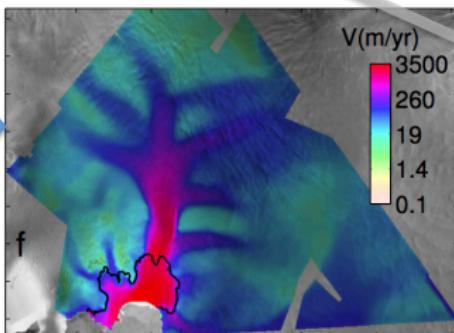
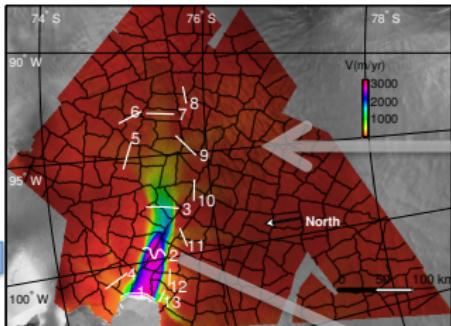
# ISSM-DAKOTA



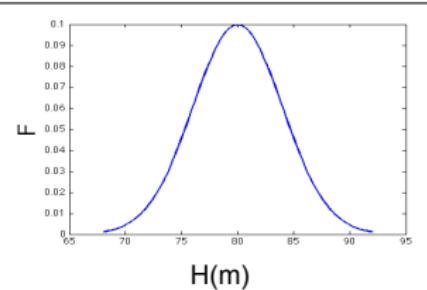
# Sampling

Repeated  
Samples

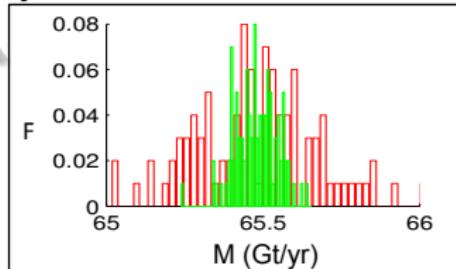
2D  
SSA



Thickness  
Errors

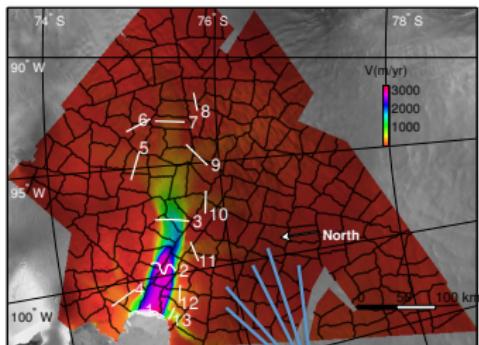


Mass Flux  
Uncertainty



# Sensitivity

Small perturbation in thickness,  
one partition at a time

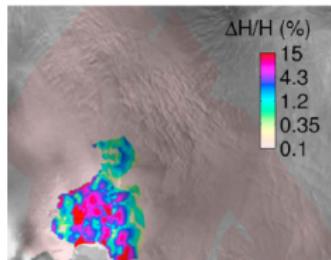


One 2D- SSA solve  
per partition

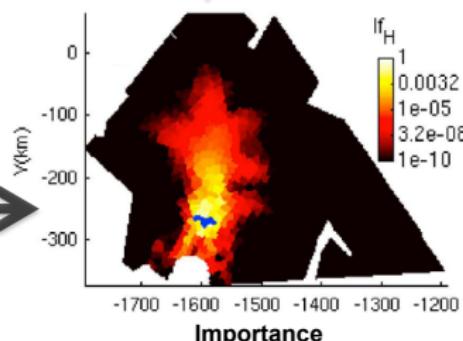
$$\begin{aligned} r &= \text{grounding line mass flux} \\ x_i &= \text{ice thickness} \end{aligned}$$

$$\theta_i = \frac{\delta r}{\delta x_i}$$

Spatial sensitivities  
of response  $r$  to  
change in variable  $x$



Thickness  
Errors



Importance  
Factors

## First Step: flux gates

Flux gates are exp files found in **Exp\_Par/MassFluxes**. The gates are positioned across PIG at the inset of tributary glaciers.

Mass fluxes will be computed (in Gt/yr) for all of these gates (using the depth-averaged ice velocity, ice thickness and ice density).

Run step 1 of the runme.m file to plot the gates overlayed over PIG surface velocities.

```
17 plotmodel(md,'data',md.results.StressbalanceSolution.Vel,'log',10,'expdisp',...
18 {'Exp_Par/MassFluxes/MassFlux1.exp','Exp_Par/MassFluxes/MassFlux2.exp',...
19 'Exp_Par/MassFluxes/MassFlux3.exp','Exp_Par/MassFluxes/MassFlux4.exp',...
20 'Exp_Par/MassFluxes/MassFlux5.exp','Exp_Par/MassFluxes/MassFlux6.exp',...
21 'Exp_Par/MassFluxes/MassFlux7.exp','Exp_Par/MassFluxes/MassFlux8.exp',...
22 'Exp_Par/MassFluxes/MassFlux9.exp','Exp_Par/MassFluxes/MassFlux10.exp',...
23 'Exp_Par/MassFluxes/MassFlux11.exp','Exp_Par/MassFluxes/MassFlux12.exp',...
24 'Exp_Par/MassFluxes/MassFlux13.exp'),...
25 'expstyle',{'k-','k-','k-','k-','k-','k-','k-',...
26 'k-','k-','k-','k-','k-','k-'},'linewidth',2,...
27 'text',{''1'', ''2'', ''3'', ''4'', ''5'', ''6'', ''7'', ...
28 ''8'', ''9'', ''10'', ''11'', ''12'', ''13''},...
29 'textposition',textpositions);
```

## Second step: load Cross-Over errors for ice thickness

For ice thickness errors, we will use McCords cross-over errors from CReSIS (courtesy of John Paden, pers. comm.). First load the errors (step2):

```
37 %load cross overs: CRESIS McCord Antarctica, 2009 (courtesy of John Paden)
38 load('./Data/CrossOvers2009.mat');
39
40 %interpolate cross over errors over our mesh vertices
41 DeltaHH=InterpFromMeshToMesh2d(index,x,y,dhh,md.mesh.x,md.mesh.y);
```

Some of these errors are too large, too small, or need to be interpolated onto a larger domain. Filter these out:

```
46 %filter out unrealistic error ranges
47 flags=ContourToNodes(md.mesh.x,md.mesh.y, 'Exp_Par/ErrorContour.exp',1);
48 pos=find(~flags); DeltaHH(pos)=0;
49
50 %avoid large unrealistic values
51 pos=find(DeltaHH>1); DeltaHH(pos)=1;
52 pos=find(DeltaHH<-1); DeltaHH(pos)=-1;
53
54 %transform into absolute errors and setup a minimum error everywhere.
55 DeltaHH=abs(DeltaHH);
56 pos=find(DeltaHH==0); pos2=find(DeltaHH≠0);
57 DeltaHH(pos)=min(DeltaHH(pos2));
```

## Third step: setup the sampling analysis

First, partition the mesh into equal area partitions. We'll start with 50.

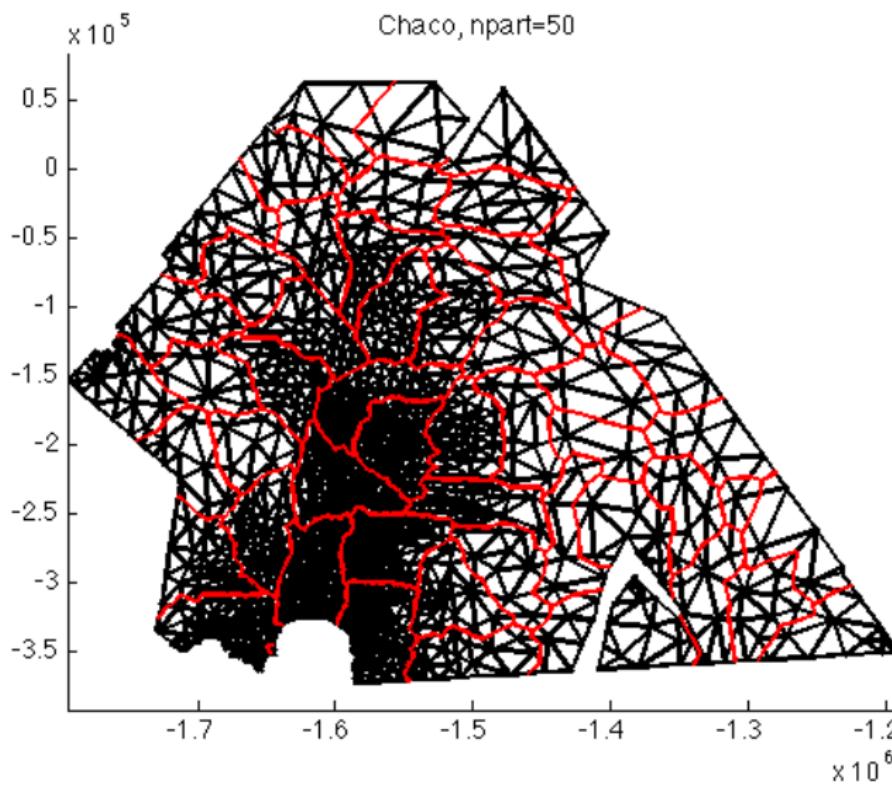
```
69 %partition the mesh
70 md.qmu.numberofpartitions=50;
71 md=partitioner(md,'package','chaco','npart',md.qmu.numberofpartitions, ...
72 'weighting','on');
73 md.qmu.partition=md.qmu.partition-1; %switch partition to c-indexing
```

You can try and play with the package for partitioning ('chaco','scotch' or 'linear'), the number of partitions, and the weighting ('on' or 'off')

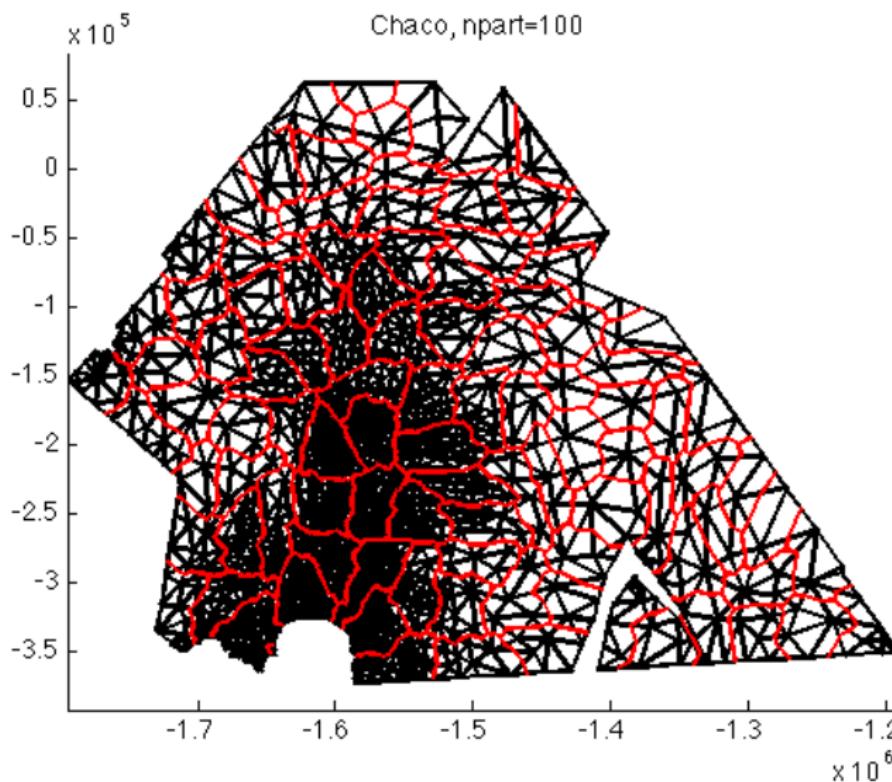
To plot the corresponding partition over a plot of the mesh:

```
229 plotmodel(md,'data','mesh','partitionedges','on','meshlinewidth',1.5, ...
230 'linewidth',2, 'axis#all','image','unit','km','colorbar','off',...
231 'title','','','meshcolor','b','grid','on');
```

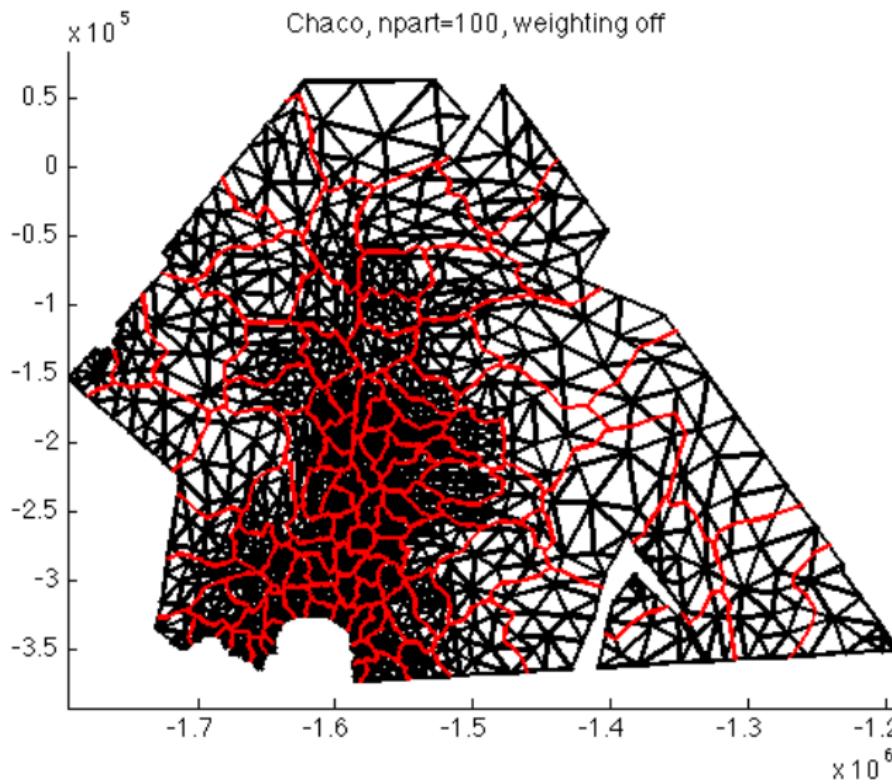
## Examples of partitions-1:



## Examples of partitions-2:



## Examples of partitions-3:



## Setup PDF Ice Thickness:

Second, setup PDF (Probability Density Function) for model input  $H$ :

```
75 %make DeltaHH into our 3 sigma deviation
76 DeltaHH=DeltaHH/6; %2 (to transform DeltaHH into a radius) x 3 (for 3 sigma)
77 DeltaHH_on_partition=AreaAverageOntoPartition(md,DeltaHH);
78 DeltaHH_on_grids=DeltaHH_on_partition(md.qmu.partition+1); %just to check ...
    in case
79
80 md.qmu.variables.thickness=normal_uncertain('scaled_Thickness',1,1);
81 md.qmu.variables.thickness.stddev=DeltaHH_on_partition;
```

Here, we need to provide the average  $\mu$  and  $\sigma$  standard deviation (from  $\Delta H/H = 6\sigma$ ) for the PDF to be defined. Then we interpolate on the partition (not the mesh!).

Anything UQ related will be in the **md.qmu** structure (qmu for Quantification of Margins and Uncertainties).

## qmu class:

Anything UQ related will be in the **md.qmu** structure (QMU stands for Quantification of Margins and Uncertainties).

```
>> md.qmu
ans =
    qmu parameters:
        isdakota      : 0          -- is qmu analysis activated?
        variables: (arrays of each variable class)
        responses: (arrays of each response class)
        numberofresponses : 0          -- number of responses
        params: (array of method-independent parameters)
        results: (information from dakota files)
        partition      : N/A        -- user provided mesh partitioning, defaults to metis if not specifi
ed
        numberofpartitions : 0          -- number of partitions for semi-discrete qmu
        variabledescriptors : (0x0)
        respondedescriptors : (0x0)
        method          : N/A        -- array of dakota_method class
        mass_flux_profile_directory: N/A
        mass_flux_profiles     : N/A        -- list of mass_flux profiles
        mass_flux_segments    : (0x0)
        adjacency           : N/A
        vertex_weight       : N/A        -- weight applied to each mesh vertex
>>
```



## Setup output diagnostics-1:

Third, setup output responses (mass flux at 13 flux gates):

```
83 %responses
84 md.qmu.responses.MassFlux1=response_function('indexed_MassFlux_1',[],...
85 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
86 md.qmu.responses.MassFlux2=response_function('indexed_MassFlux_2',[],...
87 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]); %grounding line
88 md.qmu.responses.MassFlux3=response_function('indexed_MassFlux_3',[],...
89 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
90 md.qmu.responses.MassFlux4=response_function('indexed_MassFlux_4',[],...
91 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
92 md.qmu.responses.MassFlux5=response_function('indexed_MassFlux_5',[],...
93 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
94 md.qmu.responses.MassFlux6=response_function('indexed_MassFlux_6',[],...
95 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
96 md.qmu.responses.MassFlux7=response_function('indexed_MassFlux_7',[],...
97 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
98 md.qmu.responses.MassFlux8=response_function('indexed_MassFlux_8',[],...
99 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
100 md.qmu.responses.MassFlux9=response_function('indexed_MassFlux_9',[],...
101 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
102 md.qmu.responses.MassFlux10=response_function('indexed_MassFlux_10',[],...
103 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
104 md.qmu.responses.MassFlux11=response_function('indexed_MassFlux_11',[],...
105 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
106 md.qmu.responses.MassFlux12=response_function('indexed_MassFlux_12',[],...
107 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999]);
108 md.qmu.responses.MassFlux13=response_function('indexed_MassFlux_13',[],...
109 [0.0001 0.001 0.01 0.25 0.5 0.75 0.99 0.999 0.9999);
```

## Setup output diagnostics-2:

For all responses, we specify a string identifier '*indexed\_MassFlux\_1*', and confidence intervals. We also need to specify an exp file to describe each flux gate, and a directory where to find the latter.

```
111 %mass flux profiles
112 md.qmu.mass_flux_profiles={'MassFlux1.exp',...
113                                'MassFlux2.exp',...
114                                'MassFlux3.exp',...
115                                'MassFlux4.exp',...
116                                'MassFlux5.exp',...
117                                'MassFlux6.exp',...
118                                'MassFlux7.exp',...
119                                'MassFlux8.exp',...
120                                'MassFlux9.exp',...
121                                'MassFlux10.exp',...
122                                'MassFlux11.exp',...
123                                'MassFlux12.exp',...
124                                'MassFlux13.exp'};
125 md.qmu.mass_flux_profile_directory='..../Exp_Par/MassFluxes/';
```

## Setup sampling engine:

Fourth, we need to decide on a sampling strategy:

```
127 %% sampling analysis
128 md.qmu.method      =dakota_method('nond_samp');
129 md.qmu.method(end)=dmeth_params_set(md.qmu.method(end),...
130 'seed',1234, ...
131 'samples',30, ...
132 'sample_type','lhs'); %random or lhs
```

We specify the type of method ('*nond\_samp*' for sampling or '*nond\_I*' for local reliability method/sensitivity analysis), following Dakota guidelines.

We determine the number of samples (30 for now) and also the type of sampling algorithm ('*lhs*' or '*random*').

## Some parameters:

We setup persistent parameters:

```
134 %% a variety of parameters
135 md.qmu.params.evaluation_concurrency=1;
136 md.qmu.params.analysis_driver='';
137 md.qmu.params.analysis_components='';
138 md.qmu.params.tabular_graphics_data=true;
139 md.verbose=verbose('qmu',true);
```

This includes parallel concurrency, verbosity, and data backup ('tabular\_graphics\_data').

## Solve:

We also have to tighten the solver tolerance (in order to avoid spurious sensitivities to develop).

And then solve:

```
141 md.stressbalance.restol=10^-5; %tighten tolerances for UQ analyses  
142  
143 %solve  
144 md.qmu.isdakota=1; md.inversion.iscontrol=0;  
145 md.cluster=generic('name',oshostname,'np',2);  
146 md=solve(md,StressbalanceSolutionEnum,'overwrite','y');
```

Don't forget to deactivate inversion (iscontrol=0), and to activate UQ run (isdakota=1).

Results will be in md.results.dakota and md.qmu.results

## Model Setup-1

Next, we quantify *importance factors* (sensitivities scaled by error margins) for model inputs: **ice thickness  $H$** , **basal friction  $\alpha$** , and **ice rigidity  $B$** . We specify a 5% error margin on all inputs.

*Partitioning is identical. Model outputs are identical.*

### To add model inputs:

```

163 %variables
164 md.qmu.variables.DragCoefficient=normal_uncertain(... 
165 'scaled_FrictionCoefficient',1,0.05);
166 md.qmu.variables.rheology_B=normal_uncertain(... 
167 'scaled_MaterialsRheologyB',1,0.05);
168 md.qmu.variables.Thickness=normal_uncertain('scaled_Thickness',1,0.05);

```

### To specify new sensitivity method: 'nond\_l':

```

201 %method: local reliability
202 md.qmu.method      =dakota_method('nond_l');
203 md.qmu.method(end)=dmeth_params_set(md.qmu.method(end),...
204 'output','quiet');
205
206 %parameters
207 md.qmu.params.evaluation_concurrency=1;
208 md.qmu.params.analysis_driver='';
209 md.qmu.params.analysis_components='';
210 md.qmu.params.tabular_graphics_data=false;
211
212 md.stressbalance.restol=10^-5; %tighten for qmu analyses

```

## Model Setup-2

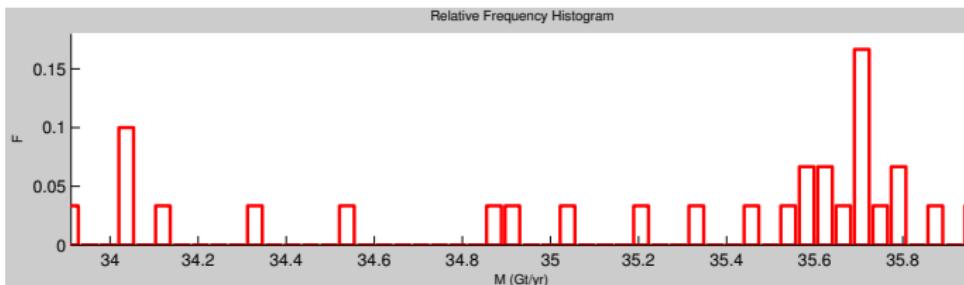
We solve the same way:

```
214 %solve
215 md.qmu.isdakota=1;
216 md.inversion.iscontrol=0;
217 md.cluster=generic('name',oshostname,'np',2);
218 md.verbose=verbose('qmu',true);
219 md=solve(md,StressbalanceSolutionEnum,'overwrite','y');
```

## Plot sampling results

In order to plot the results, we pick-up one of the flux gates, and display a histogram of the sampling results.

```
240 %which profile are we looking at?
241 index=1;
242
243 %retrieve results for the specific profile, mass flux in m^3 water equiv/s
244 result=md.results.dakota.dresp_dat(md.qmu.numberofpartitions+index);
245 result.sample=result.sample/1e12*60*60*24*365;
246
247 %plot histogram
248 plot_hist_norm(result,'cdfleg','off','cdfplt','off','nrmplt','off',...
249 ' xlabelplt','M (Gt/yr)', ' ylabelplt','F', 'FontSize',8, 'FaceColor',...
250 'none', 'EdgeColor','red');
```



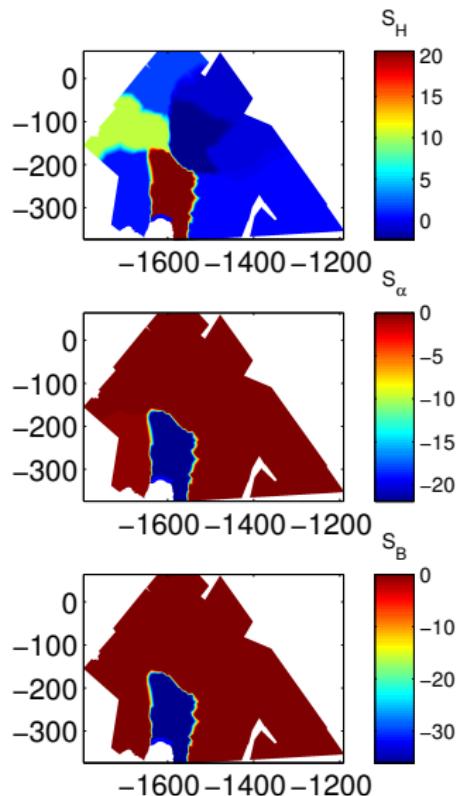
## Plot sensitivity results

To retrieve sensitivities for each model input:

```
261 %which profile are we looking at?
262 index=1;
263
264 %To plot sensitivities
265 sa=md.results.dakota.dresp_out(index).sens(1:10); ...
    sa=sa(md.qmu.partition+1)/1e12*60*60*24*365;
266 sb=md.results.dakota.dresp_out(index).sens(11:20); ...
    sb=sb(md.qmu.partition+1)/1e12*60*60*24*365;
267 sh=md.results.dakota.dresp_out(index).sens(21:30); ...
    sh=sh(md.qmu.partition+1)/1e12*60*60*24*365;
```

To plot sensitivities:

```
269 plotmodel(md,'data',sh,'data',sa,'data',sb,'expdisp#all',...
270 ['Exp_Par/MassFluxes/MassFlux' num2str(index) '.exp'],...
271 'expstyle#all','b-','linewidth#all',2,...
272 'nlines',3,'ncols',1,'axis#all','image',...
273 'colorbar#all','on','colorbarfontsize#all',10,...
274 'colorbartitle#1','S_{H}', 'colorbartitle#2','S_{\alpha}',...
275 'colorbartitle#3','S_{B}', 'unit#all','km','figure',1);
```



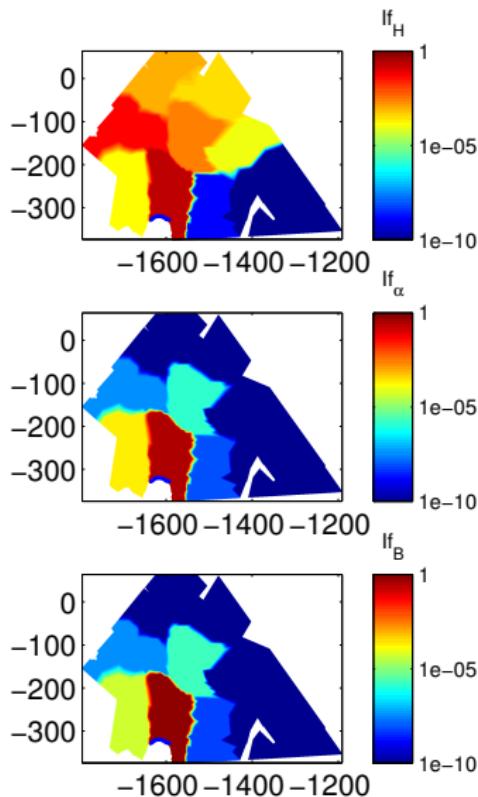
## Plot Importance Factors

To retrieve importance factors for each model input:

```
277 %To plot importance factors
278 ifa=importancefactors(md, 'scaled_FrictionCoefficient', ['indexed_MassFlux_',...
    num2str(index)]);
279 ifb=importancefactors(md, 'scaled_MaterialsRheologyB', ['indexed_MassFlux_',...
    num2str(index)]);
280 ifh=importancefactors(md, 'scaled_Thickness', ['indexed_MassFlux_',...
    num2str(index)]);
```

To plot importance factors:

```
282 plotmodel(md, 'data',ifh,'data',ifa,'data',ifb,'expdisp#all',...
283 ['Exp_Par/MassFluxes/MassFlux' num2str(index) '.exp'],...
284 'expstyle#all','b-','linewidth#all',2,'log#all',10,...
285 'nlines',3,'ncols',1, 'axis#all','image','caxis#all',[1e-10 1],...
286 'colorbar#all','on','colorbarfontsize#all',10,...
287 'colorbartitle#1','If_{H}', 'colorbartitle#2','If_{\alpha}',...
288 'colorbartitle#3','If_{B}', 'unit#all','km','figure',2);
```



## Conclusion

UQ allows the following:

- assess propagation of errors from model inputs to model diagnostics
- assess sensitivities of model diagnostics with respect to model inputs
- quantify error margins for projections

Our UQ capabilities apply to any model input, any model output, any solution, irrespective of the type of sampling or sensitivity analysis being carried out.

# QMU

## Exercises

Extra Time? Try to something more complicated.

Addditional Suggested Exercises:

- Add diagnostics IceVolume or MaxVelocity
- Sample with a uniform distribution (See help uniform\_uncertain)
- Sample additional variables (i.e. friction coefficient, ice rheology)
- Try qmu on a different solution type
- Change number of partitions, NB: for sensitivity this could take a while!

A wide-angle photograph of a desolate, icy terrain. In the foreground, a flat expanse of white, textured snow or ice stretches across the frame. Behind it, a range of mountains rises, their peaks covered in thick, white snow. The mountains are rugged, with deep shadows in the valleys and bright reflections on the snow. The sky above is a clear, pale blue, with a few wispy clouds near the horizon.

Thanks!